**Test Plan**

**<”*Wantsome*” Booking Web Application >**

**Version:** 1.0

**Created:** 04/05/2023

**Last Updated:** 04/20/2023

**Tester**: Darasteanu Razvan

**Status:** DRAFT (The status would change to finalized post the BA, PM and dev team review and sign off!)

# Revision and Signoff Sheet

## Document History - To maintain a list of changes being made

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Description of Change** |
| **1** | **04/14/2023** | **Darasteanu Razvan** | **Draft** |
| **2** | **04/20/2023** | **Darasteanu Razvan** | **Draft - Reviewed** |
|  |  |  |  |

## Approvers List - To track who has reviewed and signoff on the Test plan

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Role** | **Approver / Reviewer** | **Approval / Review Date** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

## Reference Documents - Clearly mark the document used as an input to create the test plan

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Document Name** |
| **1.0** | **04/05/2023** | **<”*Wantsome*” Booking Web Application >** |
|  |  |  |

# Table of Contents

[Revision and Signoff Sheet 2](#_Toc132997202)

[Document History - To maintain a list of changes being made 2](#_Toc132997203)

[Approvers List - To track who has reviewed and signoff on the Test plan 2](#_Toc132997204)

[Reference Documents - Clearly mark the document used as an input to create the test plan 2](#_Toc132997205)

[Table of Contents 3](#_Toc132997206)

[Chapter 1. Introduction 7](#_Toc132997207)

[1.1 Purpose 7](#_Toc132997208)

[1.2 Scope 7](#_Toc132997209)

[1.3 Document Overview 7](#_Toc132997210)

[1.4 Audience: 8](#_Toc132997211)

[a) Product Owners 8](#_Toc132997212)

[b) Project Managers 8](#_Toc132997213)

[c) Business Analysts 8](#_Toc132997214)

[d) Development team 8](#_Toc132997215)

[e) Quality Assurance team 8](#_Toc132997216)

[f) Other stakeholders 8](#_Toc132997217)

[1.5 References 8](#_Toc132997218)

[Chapter 2. Test Strategy for Booking Application Website 9](#_Toc132997219)

[2.1 Objectives 9](#_Toc132997220)

[2.2 Testing Scope 9](#_Toc132997221)

[2.3 Testing Approach 9](#_Toc132997222)

[2.4 Test Levels 10](#_Toc132997223)

[a) Unit testing 10](#_Toc132997224)

[b) Integration testing 10](#_Toc132997225)

[c) System testing 10](#_Toc132997226)

[d) User Acceptance Testing (UAT) 10](#_Toc132997227)

[2.5 Test Types 11](#_Toc132997228)

[2.5.1 Functional Testing 11](#_Toc132997229)

[a) Smoke Testing 11](#_Toc132997230)

[b) Regression Testing 11](#_Toc132997231)

[c) Sanity Testing 11](#_Toc132997232)

[d) Retesting 12](#_Toc132997233)

[2.5.2 Non-functional Testing 12](#_Toc132997234)

[a) Performance Testing 12](#_Toc132997235)

[b) Security Testing 12](#_Toc132997236)

[c) Scalability Testing 12](#_Toc132997237)

[d) Recoverability Testing 13](#_Toc132997238)

[e) Usability Testing 13](#_Toc132997239)

[f) Compatibility Testing 13](#_Toc132997240)

[g) Localization Testing 13](#_Toc132997241)

[2.6 Test Design Techniques 13](#_Toc132997242)

[a) Equivalence Partitioning 13](#_Toc132997243)

[b) Boundary Value Analysis 14](#_Toc132997244)

[c) Decision Table 14](#_Toc132997245)

[d) State Transition Testing 14](#_Toc132997246)

[e) Error Guessing 15](#_Toc132997247)

[2.7 Test Environment 15](#_Toc132997248)

[2.8 Test Data Management 15](#_Toc132997249)

[2.9 Risk Assessment and Mitigation 15](#_Toc132997250)

[a) Security risks 15](#_Toc132997251)

[b) Compatibility risks 16](#_Toc132997252)

[c) Performance risks 16](#_Toc132997253)

[d) Payment gateway risks: 16](#_Toc132997254)

[e) User interface risks: 16](#_Toc132997255)

[2.10 Assumptions and Dependencies 16](#_Toc132997256)

[Chapter 3. Test Plan Schedule 17](#_Toc132997257)

[3.1 Milestones 17](#_Toc132997258)

[3.2 Deliverables 18](#_Toc132997259)

[3.3 Test Schedule 18](#_Toc132997260)

[a) Test Planning 18](#_Toc132997261)

[b) Test Case Preparation 18](#_Toc132997262)

[c) Test Execution 19](#_Toc132997263)

[d) Bug Fixing and Retest 19](#_Toc132997264)

[e) Final Test Execution 19](#_Toc132997265)

[Chapter 4. Test Execution Strategy 20](#_Toc132997266)

[4.1 Test Execution Approach 20](#_Toc132997267)

[4.2 Test Cycles 20](#_Toc132997268)

[4.3 Test Entry and Exit Criteria 20](#_Toc132997269)

[4.4 Validation and Defect Management 20](#_Toc132997270)

[4.5 Test Progress Monitoring and Control 21](#_Toc132997271)

[4.6 Test Metrics and Measurements 21](#_Toc132997272)

[4.7 Test Reporting 21](#_Toc132997273)

[4.8 Defect Tracking and Reporting 22](#_Toc132997274)

[4.9 Change Management 22](#_Toc132997275)

[Chapter 5. Test Management 23](#_Toc132997276)

[5.1 Test Management Tools. Jira Tool 23](#_Toc132997277)

[5.2 Test Case Design 24](#_Toc132997278)

[5.3 Test Case Execution 24](#_Toc132997279)

[Chapter 6. Team Organization 25](#_Toc132997280)

[6.1 Roles and Responsibilities - An Agile Perspective 25](#_Toc132997281)

[a) Product Owner: 25](#_Toc132997282)

[b) Project Manager: 25](#_Toc132997283)

[c) Business Analysts: 25](#_Toc132997284)

[d) System Analyst: 26](#_Toc132997285)

[e) Scrum Master: 26](#_Toc132997286)

[f) Development Team: 26](#_Toc132997287)

[g) Test Manager 27](#_Toc132997288)

[h) Test Lead 28](#_Toc132997289)

[i) Quality Assurance Team 28](#_Toc132997290)

[j) Deployment and Operations Team 29](#_Toc132997291)

[k) Stakeholders 29](#_Toc132997292)

[6.2 Communication Plan 29](#_Toc132997293)

[6.3 Training and Skill Development 30](#_Toc132997294)

[Chapter 7. Test Environment 31](#_Toc132997295)

[7.1 Hardware and Software Requirements 31](#_Toc132997296)

[7.2 Test Data 31](#_Toc132997297)

[7.3 Test Tools 31](#_Toc132997298)

[7.4 Test Infrastructure 32](#_Toc132997299)

[Chapter 8. Approval 33](#_Toc132997300)

[8.1 Sign-off 33](#_Toc132997301)

[8.2 Exit Criteria 33](#_Toc132997302)

[8.3 The Names and Titles of all persons who must approve this plan 34](#_Toc132997303)

[Chapter 9. Appendix 35](#_Toc132997304)

[9.1 Test Case Template 35](#_Toc132997305)

[9.2 Test Case Execution Template 37](#_Toc132997306)

[9.3 Bug/ Defect Report Template 38](#_Toc132997307)

[9.4 Test Execution Approach Template 39](#_Toc132997308)

[9.5 Exit Criteria Template 40](#_Toc132997309)

[9.6 Test Cycles Template 41](#_Toc132997310)

[9.7 Validation and Defect Management Template 41](#_Toc132997311)

[9.8 Test Progress Monitoring and Control Template 42](#_Toc132997312)

[9.9 Test Metrics and Measurements Template 42](#_Toc132997313)

[9.10 Test Reporting Template 43](#_Toc132997314)

[9.11 Defect Tracking and Reporting 43](#_Toc132997315)

[Chapter 10. Glossary of Terms 44](#_Toc132997316)

Chapter 1. Introduction

This test plan is designed to ensure the quality of a booking application website that allows users to book hotels in various locations throughout Romania. The website is hosted at the URL <http://138.68.69.185:3333/>. The primary objective of this test plan is to identify any potential issues or bugs in the application and to ensure that it meets the requirements and expectations of its intended users.

The testing will be conducted by a team of experienced testers who will follow a set of predefined test cases and scenarios to ensure that the website is fully functional and meets all the necessary standards. The results of the testing will be documented and shared with the development team to facilitate any necessary changes and improvements to the application.

## 1.1 Purpose

The purpose of this test plan is to provide a comprehensive strategy and approach for testing the Booking Application Website for hotels in Romania with the URL <http://138.68.69.185:3333/>. The primary objective of this test plan is to ensure that the application meets all the requirements and specifications outlined by the stakeholders and is functional, reliable, and user-friendly for end-users.

## 1.2 Scope

The scope of this test plan includes testing all the functionalities and features of the Booking Application Website for hotels in Romania, including but not limited to the search and booking process, user registration and login, payment gateway integration, cancellation policy, and customer support. The testing will cover various testing types such as functional, exploratory, regression, performance, security, and user acceptance testing.

## 1.3 Document Overview

This test plan will provide an overview of the testing process and approach for the Booking Application Website for hotels in Romania. It includes the test strategy, test schedule, test execution strategy, test management, team organization, test environment, and approvals. The appendix includes test case templates, test execution templates, defect report templates, and a glossary of terms.

## 1.4 Audience:

The audience for the booking application test plan is primarily the development team, quality assurance team, and project stakeholders. This includes product owners, project managers, , business analysts, developers, testers and other stakeholders.

a) Product Owners: Product owners are responsible for ensuring that the application meets the needs of the end-users.

b) Project Managers: Project managers are responsible for managing the project timeline and ensuring that the project is completed on time and within budget.

c) Business Analysts: Business analysts are responsible for defining the business requirements for the application.

d) Development team: The developers are responsible for developing the application according to the specifications provided by the project stakeholders

e) Quality Assurance team: The testing team will be responsible for ensuring that the application works as expected and meets the requirements of the stakeholders.

f) Other stakeholders: Other stakeholders such as investors, customers, and partners may also be interested in the testing process and the results of the testing.

## 1.5 References

* Booking Application Requirements Document
* Booking Application Design Document
* Booking Application User Manual
* Hotel Industry Standards and Guidelines
* Test Case Templates
* Test Execution Templates
* Defect Report Templates

# Chapter 2. Test Strategy for Booking Application Website

## 2.1 Objectives

The primary objective of the testing strategy is to ensure the quality of the booking application website for hotels in Romania, with the URL <http://138.68.69.185:3333/>. The following objectives are set for the testing process:

* Validate the functionality of the website, including the booking process, payment gateway, and user management system.
* Ensure that the website is responsive and user-friendly, and provides an excellent user experience.
* Verify that the website is compatible with different web browsers and devices.
* Ensure that the website is secure and can handle a large number of user requests.

2.2 Testing Scope

The testing scope for the booking application website for hotels in Romania includes the following:

* Functional testing: smoke testing, regression testing, sanity testing, re-testing
* Non Functional testing: performance testing, security testing, scalability testing, recoverability testing, usability testing, compatibility testing and localization testing.

2.3 Testing Approach

The testing approach for the booking application website for hotels in Romania will follow an agile methodology, with continuous testing throughout the development cycle. The testing will be performed in parallel with development to ensure any issues are identified early in the development process.

## 2.4 Test Levels

The following test levels will be conducted for the booking application website for hotels in Romania:

a) Unit testing**:**

* To test individual components of the website.
* White box testing is typically used for unit testing**.**

b) Integration testing**:**

* To test the integration of different components of the website.
* Black box testing is commonly used for integration testing.

c) System testing**:**

* To test the complete system, including functional, compatibility, performance, and security testing.
* Grey box testing is usually used for system testing, as it involves testing the system as a whole while still having some knowledge of the internal workings of the system.

d) User Acceptance Testing (UAT)**:**

* To ensure that the website meets the expectations and requirements of its intended users.
* UAT will involve testing the website with real users who will be asked to perform specific tasks on the website. The UAT testing will be conducted by a separate team of testers who are representative of the intended users of the application. The UAT team will provide feedback on the usability, accessibility, and overall user experience of the website.
* UAT is typically performed using black box testing.

The testing results from each level will be documented, and any issues or bugs found will be reported to the development team for resolution. The testing team will continue to perform regression testing to ensure that any fixes made do not affect other parts of the application.

## 2.5 Test Types

The following test types will be conducted for the booking application website for hotels in Romania:

2.5.1 Functional Testing:

* Functional testing will be conducted to ensure that the website meets its functional requirements. The following types of functional testing will be performed:

a) Smoke Testing:

* This testing will be conducted to ensure that the critical functionalities of the website are working correctly. Smoke testing will be performed after each build to check if the website is ready for further testing. The main goal of smoke testing is to identify any major issues that would prevent further testing.

b) Regression Testing:

* Regression testing will be performed to ensure that the changes made to the website do not impact the existing functionality of the application. Regression testing will be conducted after each build to verify that previously working functionalities have not been affected by the new changes.

c) Sanity Testing:

* Sanity testing will be conducted to ensure that the modified functionalities or new features work as expected. Sanity testing is performed after the smoke test and before the regression test to ensure that the application is ready for further testing.

d) Retesting:

* Retesting will be performed to ensure that the defects reported earlier have been fixed and are working as expected. Retesting will be conducted after each bug fix to verify that the issue has been resolved.

2.5.2 Non-functional Testing:

Apart from functional testing, the following non-functional testing will be conducted for the booking application:

a) Performance Testing:

* The performance of the website will be tested under various conditions, such as peak traffic, average traffic, and low traffic. The aim of performance testing is to ensure that the website functions optimally, with minimal load times and response times, and to identify any bottlenecks in the system. The performance testing will be conducted using tools like JMeter and LoadRunner.

b) Security Testing:

* The security of the website will be tested to ensure that the website is secure against any unauthorized access or attacks. This testing will include vulnerability scanning, penetration testing, and authentication testing. The testing will be conducted using tools like OWASP ZAP, Nessus, and Burp Suite.

c) Scalability Testing:

* The scalability of the website will be tested to ensure that the website can handle increased traffic, users, and data. The testing will be conducted by gradually increasing the traffic to the website and monitoring its performance. This testing will help identify the website's ability to scale horizontally and vertically.

d) Recoverability Testing:

Recoverability testing will be conducted to ensure that the website can recover from any failures or disruptions. The testing will include backup and recovery testing, disaster recovery testing, and failover testing.

e) Usability Testing:

* Usability testing will be conducted to ensure that the website is user-friendly and easy to navigate. The testing will be conducted with real users to get feedback on the website's layout, design, and overall user experience. This testing will help identify any issues with the website's interface and ensure that the website is accessible to all users.

f) Compatibility Testing:

* Compatibility testing will be conducted to ensure that the website is compatible with different devices, browsers, and operating systems. The testing will be conducted using tools like BrowserStack and Sauce Labs to test the website's compatibility with different configurations.

g) Localization Testing:

* This testing is performed to ensure that the website is compatible with different languages and cultures. This testing will involve testing the website with different languages and verifying that the website behaves consistently across different cultures.

## 2.6 Test Design Techniques

The following test techniques will be used for testing the booking application website for hotels in Romania:

a) Equivalence Partitioning:

* This technique involves testing the system by dividing input data into equivalent groups and testing each group. Testers will identify different groups of input data that are treated in the same way by the application and test each group to ensure that the system handles the input data correctly. The aim of this testing is to ensure that the website behaves consistently for input data that falls within the same equivalence class.

b) Boundary Value Analysis:

* This technique involves testing the system with minimum and maximum values to validate the system's behavior at the extremes. Testers will test the website with values that are at the boundary of valid input data. The aim of this testing is to ensure that the website can handle values that are close to the limit of the application's capabilities and to identify any issues or bugs that may occur at the limit.

c) Decision Table:

* This technique involves creating a matrix that outlines all possible combinations of inputs and outputs for a given feature or function of the application. Testers will use this matrix to ensure that all possible combinations have been tested and the system behaves correctly for each combination. The aim of this testing is to identify any issues or bugs that may occur due to the interaction between different inputs.

d) State Transition Testing:

* This technique involves testing the application's behavior when moving from one state to another. Testers will identify the different states of the application and test the application's behavior as it transitions from one state to another. The aim of this testing is to identify any issues or bugs that may occur during the transition between states, such as the loss of data or incorrect application behavior.

e) Error Guessing:

* This technique involves anticipating and testing common errors that may occur in the system. Testers will use their knowledge and experience to identify potential errors in the system and test them to ensure that the application handles errors correctly. The aim of this testing is to identify and fix any issues or bugs that may occur due to common user errors.

## 2.7 Test Environment

The following test environment will be used for testing the booking application website for hotels in Romania:

* Operating System: Windows and macOS
* Web Browsers: Google Chrome, Mozilla Firefox, Safari, and Microsoft Edge
* Devices: Desktop and mobile devices
* Test Management Tool: JIRA
* Test Automation Tool: Selenium WebDriver

2.8 Test Data Management:

The test data for the booking application website for hotels in Romania will be managed using a test data management tool. The test data will be created to cover various test scenarios, including positive and negative testing.

2.9 Risk Assessment and Mitigation:

The following risks have been identified and will be mitigated during the testing process:

a) Security risks:

* Unauthorized access and data breaches, will be mitigated by performing regular security testing and implementing appropriate security measures.
* There is a risk of unauthorized access to the website and customer data. Mitigation: Implementing appropriate security measures like SSL certificate, access control, and encryption.

b) Compatibility risks:

* The website's incompatibility with different web browsers and devices, will be mitigated by conducting compatibility testing and implementing appropriate compatibility measures

c) Performance risks:

* The website may experience high traffic during peak periods which may affect its performance. Mitigation: Conducting load testing to determine the website's capacity and implementing measures to optimize performance.
* The website's inability to handle a large number of user requests, will be mitigated by conducting performance testing and implementing appropriate performance optimization techniques.

### **d) Payment gateway risks**:

* The payment gateway may have vulnerabilities that could result in fraudulent transactions. Mitigation: Selecting a reliable payment gateway provider and conducting regular security checks.

### **e) User interface risks**:

* The user interface may not be user-friendly, which could lead to errors during booking. Mitigation: Conducting user testing and implementing feedback to improve the user interface.

2.10 Assumptions and Dependencies:

* The website has been developed and deployed successfully on the server.
* The database is functioning properly and can store and retrieve data accurately.
* The payment gateway is integrated with the website and can process transactions successfully.
* Users have access to a reliable internet connection and a compatible web browser to access the website.
* The website complies with all applicable laws and regulations in Romania, including data protection and privacy laws.

# Chapter 3. Test Plan Schedule

The Test Plan Schedule outlines the timeline for testing the application to ensure it meets the requirements and specifications. It includes Milestones, Deliverables, and Test Schedule.

## 3.1 Milestones

Milestones are the significant events or checkpoints during the testing process. The milestones for the Test Plan Schedule can include:

**a)** Completion of the test plan:

* The test plan should be completed and reviewed by the stakeholders before the testing starts.

**b)** Completion of the test cases:

* The test cases should be created and reviewed by the testing team.

**c)** Completion of the testing:

* The testing should be completed for all the functionalities and features of the application.

**d)** Completion of bug fixes:

* The bugs should be fixed and verified by the testing team before the final delivery.

## 3.2 Deliverables

Deliverables are the tangible outputs of the testing process. The deliverables for the Test Plan Schedule can include:

**a)** Test plan document:

* The test plan document should be created and reviewed by the stakeholders.

**b)** Test cases document:

* The test cases document should be created and reviewed by the testing team.

**c)** Test execution report:

* The test execution report should be created and reviewed by the testing team, which includes the test results and bugs found during testing.

**d)** Bug report document:

* The bug report document should be created and reviewed by the testing team, which includes all the bugs found during testing.

3.3 Test Schedule

The test schedule includes the timeline for executing the test cases and achieving the milestones. It can include the following phases:

a) Test Planning:

* This phase includes the creation of the test plan document, identifying the testing scope, and preparing the test environment.

b) Test Case Preparation:

* This phase includes the creation of the test cases document based on the requirements and specifications.

c) Test Execution:

* This phase includes the execution of the test cases, identifying and reporting the bugs, and verifying the fixes.

d) Bug Fixing and Retest:

* This phase includes fixing the reported bugs, retesting the fixed bugs, and updating the test execution report.

e) Final Test Execution:

* This phase includes the final testing of the application before the delivery.

The Test Schedule can also include the following timelines:

* Test Planning: 2 days
* Test Case Preparation: 5 days
* Test Execution: 10 days
* Bug Fixing and Retest: 5 days
* Final Test Execution: 2 days

The above timelines are subject to change depending on the complexity of the application and the number of test cases to be executed.

In summary, the Test Plan Schedule is critical in ensuring that the testing process is well-structured and executed within the timelines. It helps to identify the milestones, deliverables, and test schedule to ensure the application meets the requirements and specifications.

# Chapter 4. Test Execution Strategy

## 4.1 Test Execution Approach

The Test Execution Approach for the booking application website will be a combination of manual and automated testing. Manual testing will be conducted to verify the user interface, functionality, usability, and compatibility of the website across different devices and browsers. Automated testing will be used to test the performance, security, and scalability of the website.

4.2 Test Cycles

The Test Execution for the booking application website will be divided into several test cycles. Each test cycle will consist of a set of test cases that will be executed to ensure the website meets the functional and non-functional requirements. Test cycles will be designed to cover all possible scenarios, including positive and negative testing, to ensure the website is robust and user-friendly.

4.3 Test Entry and Exit Criteria

The Test Entry Criteria for the booking application website will include the completion of the development phase, code reviews, and the availability of the test environment. The Test Exit Criteria will include the successful completion of all test cycles, bug resolution, and the acceptance of the website by the stakeholders.

## **4.4 Validation and Defect Management**

Validation and Defect Management are critical aspects of any testing effort, and are essential to ensuring the quality of the booking application website. The goal of validation is to verify that the website meets the specified requirements and is free from defects. Defect management, on the other hand, involves tracking and managing defects found during testing to ensure they are resolved before the website is released to end-users.

Here are the steps of the process:

* Testers execute scripts in each cycle and may do additional testing if they find gaps.
* Defects are logged in Jira and tracked by the technical team.
* Testers open, link, assign initial severity and status, retest and close defects.
* Defect Manager reviews severity, facilitates fix and implementation, communicates with testers, requests retests, and modifies status.
* Technical team reviews Jira, fixes defects, communicates with Defect Manager, and implements solutions.

4.5 Test Progress Monitoring and Control

Test Progress Monitoring and Control will be an essential part of the Test Execution Strategy for the booking application website. Test progress will be monitored using metrics such as test case execution status, defect density, and test coverage. A Test Manager will oversee the testing process and ensure that the testing stays on track and within the defined timelines. Test reports will be generated regularly to track the progress and identify any deviations from the plan.

4.6 Test Metrics and Measurements:

Various test metrics and measurements will be captured during the test execution. These include the number of test cases executed, passed, failed, blocked, and deferred. Other metrics such as test coverage, defect density, and test effectiveness will also be captured to measure the overall quality of the application.

Also, test metrics are used to measure the progress and effectiveness of the testing effort.

## 4.7 Test Reporting

Test Reporting will be an essential component of the Test Execution Strategy for the booking application website. Test reports will be generated after each test cycle and shared with the stakeholders. Test reports will include metrics such as test coverage, test case execution status, defect density, and test progress. The Test Manager will use these reports to identify areas for improvement and take corrective actions if required.

## 4.8 Defect Tracking and Reporting

Defects will be tracked using Jira tool, and regular defect status reports will be provided to the project stakeholders. Defect reports will include the number of open defects, the number of defects resolved, and the number of defects that have been deferred to a future release.

Defects will be logged in Jira by the testers during the test execution. The defects will be reviewed and prioritized by the Defect Manager, who will work with the technical team to fix and implement the solutions. The defect status will be updated in Jira throughout the defect management process.

## 4.9 Change Management

Any changes to the project scope, requirements, or schedule will be communicated to the relevant stakeholders through Confluence. The Change Management Board will review and approve the changes before they are implemented. The impact of the changes on the test plan and test cases will be assessed, and the necessary updates will be made. The updated test plan and test cases will be communicated to the relevant stakeholders.

# Chapter 5. Test Management

## 5.1 Test Management Tools. Jira Tool

The test management tool used for this project is Jira for defect tracking and management. Jira will be used to track all issues found during the testing process, and to manage the testing process itself.

Here are some of the things that can be done in Jira:

* Defect tracking and management: Jira can be used to log, track and manage defects found during testing. Defects can be logged as issues in Jira, with relevant information such as issue description, steps to reproduce, and screenshots or videos of the issue.
* Test case management: Jira can be used to manage and organize test cases. Test cases can be created as issues in Jira, with all relevant information such as test case steps, expected results, and associated data included in the issue description.
* Test cycle management: Jira can be used to manage test cycles, which are a set of test cases executed together as part of a testing effort. Jira can be used to track the status of each test case in the test cycle, and to manage the progress of the testing effort.
* Test planning: Jira can be used to plan and organize testing efforts. Test plans can be created in Jira, which outline the testing objectives, scope, and strategy for a specific release or iteration.
* Traceability management: Jira can be used to manage traceability between requirements, test cases, and defects. Jira can be used to link test cases to requirements, and defects to test cases, to ensure that all requirements are tested and all defects are tracked and resolved.
* Reporting: Jira can be used to generate various reports related to testing, such as defect trend analysis, test case execution status, and overall testing progress.
* Integration: Jira can be integrated with various testing tools, such as automated testing tools or performance testing tools, to streamline the testing process and improve efficiency.

5.2 Test Case Design

The test cases will be designed based on the requirements and acceptance criteria provided by the client. The test cases will cover various scenarios related to hotel booking, cancellation, modification, payment, and other functionalities of the application. The test cases will be designed to cover positive and negative scenarios to ensure the application functions as expected.

The test cases will be created as issues in Jira, with all relevant information such as the test case steps, expected results, and any associated data included in the issue description.

## 5.3 Test Case Execution

The test cases will be executed by the testers based on the test cycles defined in the test plan. The testers will use Jira to access and execute the test cases.

The execution of each test case will be tracked by updating the corresponding Jira issue with the actual results and any defects found during the execution. Any new defects identified during the execution will be logged as new issues in Jira, with all relevant details and steps to reproduce included in the issue description.

Jira will also be used to track the overall progress of the testing effort, including the number of test cases executed, the number of defects found, and the overall status of the testing effort. This information will be used to make informed decisions about the readiness of the application for release.

Chapter 6. Team Organization

6.1 Roles and Responsibilities - An Agile Perspective

The team for this project consists of the following roles and responsibilities:

### a) Product Owner:

* Defines and prioritizes the product backlog (a list of all the features and requirements for the software)
* Works with the development team to ensure that the highest priority items are worked on first
* Answers questions and provides feedback to the development team as needed
* Reviews and accepts completed work from the development team
* Ensures that the software meets the customer or user's needs and provides business value

### **b) Project Manager**:

* Oversees the entire software development project from start to finish
* Develops and manages the project schedule and budget
* Communicates with stakeholders to keep them informed of the project's progress
* Identifies and manages project risks
* Ensures that the project is completed on time, within budget, and to the required quality standards

### **c) Business Analysts**:

* Responsible for participating in the test execution during the second cycle to provide insight into the business processes.
* Identifies and gathers business requirements from stakeholders
* Analyzes business requirements to identify functional and non-functional requirements
* Documents business requirements in a clear and concise manner
* Communicates with the development team to ensure that the business requirements are fully understood

### **d) System Analyst**:

* Analyzes and designs technical requirements for the system
* Works with the business analyst to ensure that the system design meets the business requirements
* Develops technical specifications for the development team to follow
* Provides technical guidance and support to the development team as needed

### **e) Scrum Master**:

* Facilitates the Scrum process by ensuring that the Scrum framework is followed correctly
* Helps the team understand and implement Agile principles and values
* Removes any obstacles or roadblocks that are preventing the team from working efficiently
* Facilitates team meetings, such as daily stand-ups, sprint planning, sprint review, and sprint retrospective
* Helps the team prioritize the product backlog and ensures that the highest priority items are worked on first
* Protects the team from outside distractions and interruptions that could negatively impact their work
* Helps the team to continuously improve their processes and practices
* Acts as a coach and mentor to the team, providing guidance and support as needed
* Facilitates communication between the team and stakeholders
* Monitors progress and helps the team to stay on track to meet their goals and deadlines
* Encourages collaboration and teamwork within the team and across departments or organizations, if necessary.

### **f) Development Team**:

* Responsible for fixing defects and implementing changes as required.
* Participate in code reviews
* Fix defects identified by the test team
* Provide support to the QA team in defect resolution
* Develops the software according to the business and technical requirements
* Designs, codes, tests, and debugs software applications
* Collaborates with other members of the team to ensure that the software meets the required quality standards
* Provides support and maintenance to the software after it has been deployed

g) Test Manager:

* Develops and executes the overall testing strategy and test plan for the project, in collaboration with stakeholders
* Determines the appropriate testing methodologies, tools, and resources to be used based on project requirements
* Allocates resources and budgets for testing activities, and ensures that the testing effort is cost-effective and efficient
* Defines and tracks testing metrics, and reports on testing progress and results to senior management and other stakeholders
* Manages the testing team, provides guidance and support to testers, and ensures that they have the necessary tools and resources to perform their testing duties effectively
* Communicates with stakeholders, such as the project manager, development team, and business analysts, to provide updates on testing progress and any issues or concerns
* Conducts risk assessments and ensures that risks associated with testing are managed appropriately
* Provides input on product quality, readiness, and release decisions, based on the testing results and quality standards
* Develops and maintains relationships with vendors and other external parties to ensure that testing resources and tools are available as needed.

h) Test Lead:

Develops the overall testing strategy and test plan for the project, in collaboration with the Test Manager and other stakeholders

* Determines the scope of testing and defines the testing approach and methodologies to be used, based on project requirements and constraints
* Identifies and manages risks associated with testing, and develops contingency plans as needed
* Determines the appropriate test environment and test data requirements, and ensures that they are available and configured correctly
* Manages the testing team, assigns tasks, and monitors progress, and ensures that the testing effort is on track and aligned with the project goals
* Provides guidance and support to testers, and ensures that they have the necessary tools and resources to perform their testing duties effectively
* Communicates with stakeholders, such as the project manager, development team, and business analysts, to provide updates on testing progress and any issues or concerns
* Conducts regular meetings with the testing team to review progress, identify issues, and plan for future testing activities
* Develops and executes test cases, test scripts, and other testing documentation, and ensures that they are aligned with the testing strategy and standards
* Reports and tracks defects, issues, and other testing results, and ensures that they are documented and addressed appropriately
* Ensures that all testing is completed on time and within budget, and provides regular updates to the Test Manager on testing progress and issues.

i) Quality Assurance Team:

* Responsible for creating and executing test plans, test cases, logging defects, retesting defects, and communicating with the Test Lead.
* Execute test cases according to test plan and test strategy
* Provide test execution status updates to the Test Lead
* Identify and communicate risks to the Test Lead
* Tests the software to identify and fix any defects or issues
* Reports and tracks defects and issues found during testing
* Provides feedback to the development team on the software's quality and usability

j) Deployment and Operations Team:

* Deploys the software to the production environment
* Monitors the software to ensure that it is running smoothly
* Provides ongoing maintenance and support to the end-users
* Addresses any issues or problems that arise with the software in a timely and efficient manner

k) Stakeholders:

* End-users: Individuals who will be using the software
* Customers: Organizations or individuals who have purchased the software
* Sponsors: Individuals or groups who have provided funding for the software development project
* Other stakeholders: Individuals or groups who have an interest in the software development project's outcome (e.g., regulatory bodies, industry associations)

6.2 Communication Plan:

The communication plan for this project includes regular team meetings, status reports, and communication through Jira. The test manager will provide regular updates to the project manager and other stakeholders as required.

6.3 Training and Skill Development:

The test team will be provided with the necessary training and skill development to effectively execute the test plan. This includes training on the test management tools, test case design, execution, and defect management. The team will also be provided with training on any relevant standards and regulations such as GDPR (General Data Protection Regulation) or PCI DSS (Payment Card Industry Data Security Standard). Ongoing skill development will be provided as required to ensure the team is equipped with the necessary skills to execute the testing activities effectively.

Chapter 7. Test Environment

7.1 Hardware and Software Requirements:

The hardware and software requirements for the Test Environment are as follows:

* Operating System: Windows, Mac, Linux
* Web Browser: Google Chrome, Mozilla Firefox, Safari, Microsoft Edge
* Internet Connectivity: Stable Internet Connection with good bandwidth
* Screen Resolution: 1366 x 768 pixels or higher
* RAM: Minimum 4GB

7.2 Test Data:

The Test Data should include real-time data for various hotels from Romania. The Test Data should include the following information:

* Hotel names, addresses, phone numbers, and email addresses
* Room types, availability, and pricing
* Guest details including name, address, phone number, email address, and payment information
* Reservation details including check-in and check-out dates, room type, number of guests, and payment information

7.3 Test Tools:

The following Test Tools will be used for the testing of the booking application website:

* Jira for Test Management and Defect Tracking
* Selenium WebDriver for Test Automation
* OWASP ZAP (Zed Attack Proxy) for Security Testing
* LoadNinja for Performance Testing

7.4 Test Infrastructure:

The Test Infrastructure will be set up in the following way:

* The Test Environment will be set up on a separate server from the Production Environment
* The Test Environment will be configured to mimic the Production Environment as closely as possible
* The Test Environment will be regularly updated with the latest build from the Development Team
* The Test Environment will be monitored for stability and performance throughout the testing phase.

# Chapter 8. Approval

8.1 Sign-off:

Before releasing the application, it is important to obtain sign-off from the stakeholders to ensure that all the requirements have been met and that the application is ready for deployment. The sign-off is a formal approval that confirms that the application has met all the requirements and is ready for release. The stakeholders who will provide the sign-off may vary depending on the organization, but typically include the project manager, business analyst, and product owner.

To obtain sign-off, the stakeholders will review the test results, verify that all issues have been resolved, and confirm that the application meets the requirements and specifications. They will then provide formal approval to release the application for deployment. Obtaining sign-off is an important step in the testing process as it ensures that the application is ready for use and meets the needs of the stakeholders.

8.2 Exit Criteria:

The exit criteria define the conditions that must be met for the application to be considered acceptable by the stakeholders. The exit criteria are documented and agreed upon by the project team, business analysts, and product owner at the beginning of the testing process. The criteria will typically include a list of features, functions, and performance requirements that the application must meet to be accepted.

The exit criteria are used to measure the success of the testing process and to ensure that the application meets the needs of the stakeholders. The criteria should be specific, measurable, achievable, relevant, and time-bound. They should also be prioritized based on the needs of the stakeholders.

8.3 The Names and Titles of all persons who must approve this plan.

|  |  |
| --- | --- |
| **Signature:** |  |
| **Name:** |  |
| **Role:** |  |
| **Date:** |  |

|  |  |
| --- | --- |
| **Signature:** |  |
| **Name:** |  |
| **Role:** |  |
| **Date:** |  |

|  |  |
| --- | --- |
| **Signature:** |  |
| **Name:** |  |
| **Role:** |  |
| **Date:** |  |

Chapter 9. Appendix

9.1 Test Case Template

The test case template is used to document the test cases that will be executed during the testing phase.

Example:

| **Test Case ID** | **Test Case Description** | **Test Steps** | **Expected Result** | **Actual Result** | **Pass/Fail** |
| --- | --- | --- | --- | --- | --- |
| TC001 | Verify homepage loads | 1. Navigate to homepage URL<br>2. Verify page title<br>3. Verify page content is displayed correctly | Homepage is displayed correctly |  |  |
| TC002 | Verify search functionality | 1. Enter valid search criteria<br>2. Click search button<br>3. Verify search results are displayed correctly | Search results are displayed correctly |  |  |
| TC003 | Verify booking process | 1. Enter valid booking details<br>2. Click book button<br>3. Verify booking confirmation page is displayed<br>4. Verify booking details are correct | Booking confirmation page is displayed with correct details |  |  |

Another example:

| **Field** | **Description** |
| --- | --- |
| **Test Case ID** | A unique identifier for the test case. |
| **Test Case Title** | A descriptive title that summarizes the objective of the test case. |
| **Test Case Description** | A brief description of the purpose of the test case. |
| **Preconditions** | Any necessary conditions that must be met before the test case can be executed. |
| **Test Steps** | A series of steps that describe the actions to be taken to execute the test case. |
| **Expected Results** | The expected outcome or result of each step in the test case. |
| **Actual Results** | The actual outcome or result of each step in the test case. |
| **Pass/Fail** | Whether the test case passed or failed. |
| **Defects/Issues** | Any defects or issues identified during the execution of the test case. |
| **Tester Name** | The name of the tester who executed the test case. |
| **Test Date** | The date on which the test case was executed. |
| **Test Environment** | The specific environment (e.g. browser, operating system) in which the test case was executed. |

9.2 Test Case Execution Template

The test case execution template is used to record the results of the test case execution.

Example:

|  |
| --- |
|  |
| **Test Case ID** | **Test Case Description** | **Test Steps** | **Expected Result** | **Actual Result** | **Pass/Fail** |
| TC001 | Verify homepage loads | 1. Navigate to homepage URL<br>2. Verify page title<br>3. Verify page content is displayed correctly | Homepage is displayed correctly | Pass |  |
| TC002 | Verify search functionality | 1. Enter valid search criteria<br>2. Click search button<br>3. Verify search results are displayed correctly | Search results are displayed correctly | Fail | Bug ID 001 |
| TC003 | Verify booking process | 1. Enter valid booking details<br>2. Click book button<br>3. Verify booking confirmation page is displayed<br>4. Verify booking details are correct | Booking confirmation page is displayed with correct details | Pass |  |

Another example:

| **Field** | **Description** |
| --- | --- |
| Test Case ID | A unique identifier for the test case. |
| Test Case Title | A descriptive title that summarizes the objective of the test case. |
| Test Suite | The group or suite of test cases to which this test case belongs. |
| Test Cycle | The specific test cycle or phase in which the test case is executed. |
| Test Data | The specific data or inputs used in the execution of the test case. |
| Test Result | The outcome or result of the test case (pass, fail, or other). |
| Defects/Issues | Any defects or issues identified during the execution of the test case. |
| Tester Name | The name of the tester who executed the test case. |
| Test Date | The date on which the test case was executed. |
| Test Environment | The specific environment (e.g. browser, operating system) in which the test case was executed. |

9.3 Bug/ Defect Report Template

Example

| **Defect ID** | **Description** | **Severity** | **Status** | **Detected By** | **Assigned To** | **Date Detected** | **Date Fixed** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 001 | Search results not displaying correctly | Major | Open | Tester A | Technical Team | 2023-04-15 | - |

Another example:

| **Field** | **Description** |
| --- | --- |
| Title | Title of the bug/defect report |
| ID | Unique identifier assigned to the bug/defect |
| Description | A detailed description of the bug/defect, including steps to reproduce |
| Severity | The level of impact on the system (e.g., critical, high, medium, low) |
| Priority | The order in which the bug/defect should be fixed, relative to other bugs/defects |
| Status | The current status of the bug/defect (e.g., open, in progress, resolved, closed) |
| Assignee | The person assigned to fix the bug/defect |
| Reporter | The person who reported the bug/defect |
| Date | The date the bug/defect was reported |
| Environment | The operating system, browser, and device used to reproduce the bug/defect |
| Attachments | Any relevant screenshots, logs, or other files that provide additional context |
| Comments | Any additional notes or comments about the bug/defect |

## 9.4 Test Execution Approach Template

Example:

|  |  |
| --- | --- |
| **Test Type** | **Description** |
| **Manual Testing** | Verification of user interface, functionality, usability, and compatibility of the website across different devices and browsers. |
| **Automated Testing** | Testing the performance, security, and scalability of the website. |

## 9.5 **Exit Criteria Template**

|  |  |
| --- | --- |
| **Criteria** | **Description** |
| **Exit Criteria**:  Functional Requirements | All functional requirements must be tested and approved by stakeholders. |
| **Exit Criteria**:  Non-functional Requirements | All non-functional requirements such as performance, security, and usability must be tested and approved by stakeholders. |
| **Exit Criteria**:  Defect Resolution | All defects must be resolved and retested to ensure that they have been fixed. |
| **Exit Criteria**:  Acceptance Criteria | The website must meet the acceptance criteria defined by the stakeholders. |
| **Exit Criteria**:  Regression Testing | All regression testing must be completed to ensure that new features or fixes have not affected existing functionality. |
| **Exit Criteria**:  Test Documentation | All test documentation must be updated to reflect the final state of the website. |
| **Exit Criteria**:  Sign-off | The website must be signed off by the stakeholders, indicating their acceptance of the website. |
| **Exit Criteria**:  Performance Benchmarks | The website must meet the performance benchmarks defined in the requirements. |
| **Exit Criteria**:  Compliance Requirements | The website must meet any compliance requirements such as GDPR (General Data Protection Regulation) or PCI DSS (Payment Card Industry Data Security Standard). |
| **Exit Criteria**:  Security Standards | The website must meet security standards such as OWASP Top 10, which is the list of the top 10 web application security risks identified by the Open Web Application Security Project. |
| **Exit Criteria**:  User Acceptance Testing | User acceptance testing must be completed to ensure that the website meets the needs of end-users. |

9.6 Test Cycles Template

|  |  |
| --- | --- |
| **Test Cycle** | **Description** |
| **Cycle 1** | Testing of the basic functionality of the website, including user login, search, and booking. |
| **Cycle 2** | Testing of advanced functionality such as payment gateway integration and cancellation of bookings. |
| **Cycle 3** | Testing of the website's compatibility with different browsers and devices. |
| **Cycle 4** | Performance and load testing to verify the website's scalability. |

## **9.7 Validation and Defect Management Template**

Defects found during the testing will be categorized like this in Jira :

| Severity | Impact |
| --- | --- |
| Blocker | This bug is critical enough to crash the system, cause file corruption, or cause potential data loss. It causes an abnormal return to the operating system (crash or a system failure message appears). It causes the application to hang and requires re-booting the system. |
| Critical | ▪ It causes a lack of vital program functionality with workaround. |
| Major | ▪ This Bug will degrade the quality of the system. However there is an intelligent workaround for achieving the desired functionality - for example through another screen. This bug prevents other areas of the product from being tested. However other areas can be independently tested. |
| Minor | ▪ There is an insufficient or unclear error message, which has minimum impact on product use. |
| Trivial | ▪ There is an insufficient or unclear error message that has no impact on product use. |

## 9.8 **Test Progress Monitoring and Control Template**

Example:

|  |  |
| --- | --- |
| **Metric** | **Description** |
| **Test Case Execution Status** | Monitoring the status of test cases, including pass, fail, blocked, and untested. |
| **Defect Density** | Calculating the number of defects found per test case or per unit of time. |
| **Test Coverage** | Measuring the percentage of the application that has been tested. |
| **Test Progress** | Monitoring the overall progress of testing against the plan. |
| **Test case execution status** | This metric measures the percentage of test cases that have been executed and passed. |
| **Defect rejection rate** | This metric measures the percentage of defects that have been rejected by the development team. |

## 9.9 Test Metrics and Measurements Template

Example:

|  |  |
| --- | --- |
| **Test Metrics and Measurements** |  |
| Number of test cases executed |  |
| Number of test cases passed |  |
| Number of test cases failed |  |
| Number of test cases blocked |  |
| Number of test cases deferred |  |
| Test coverage |  |
| Defect density |  |
| Test effectiveness |  |

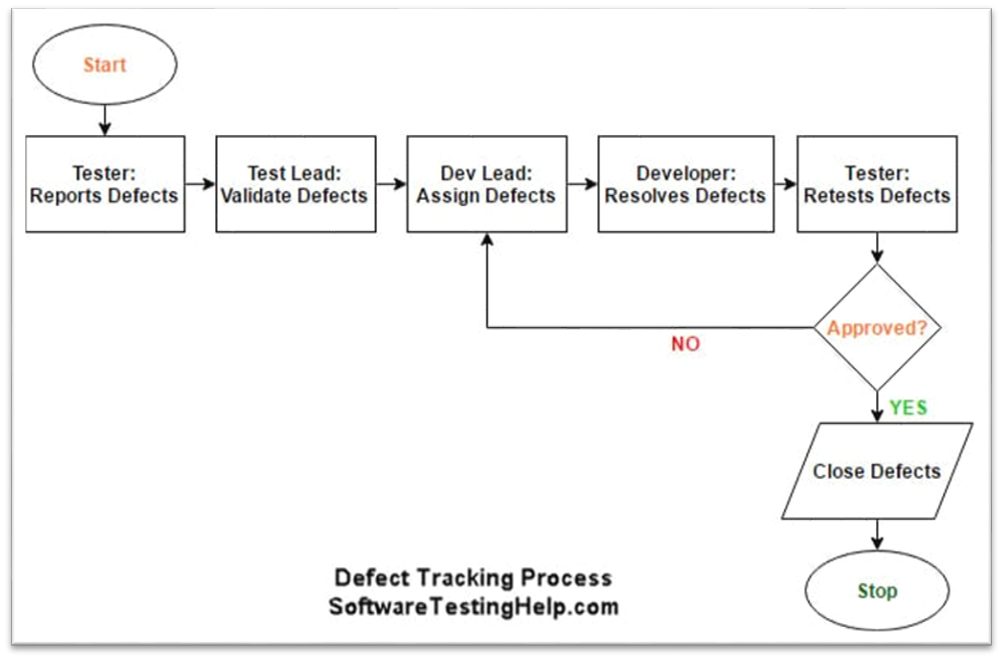
## 9.10 **Test Reporting Template**

Example:

|  |  |
| --- | --- |
| **Report** | **Description** |
| **Test Coverage Report** | Provides an overview of the percentage of the application that has been tested, broken down by module or feature. |
| **Test Execution Report** | Provides an overview of the status of test cases, including pass, fail, blocked, and untested. |
| **Defect Report** | Provides details of all defects found during testing, including severity, priority, and steps to reproduce. |
| **Test Progress Report** | Provides an overview of the overall progress of testing against the plan, highlighting any deviations or delays. |

## 9.11 Defect Tracking and Reporting

Example:



# Chapter 10. Glossary of Terms

The glossary of terms provides definitions of the technical terms and acronyms used throughout the test plan.

* Test Plan - A document that outlines the approach, objectives, scope, and schedule of testing for a software application.
* Test Strategy - A high-level plan that outlines the approach, objectives, and scope of testing for a software application.
* Test Execution Strategy - A plan that outlines the approach for executing the tests, including test cycles, entry and exit criteria, progress monitoring, and reporting.
* Test Management - The process of planning, executing, monitoring, and controlling the testing of a software application.
* Defect Management - The process of identifying, reporting, prioritizing, and resolving defects or issues found during testing.
* Change Management - The process of managing changes to the software application, including changes to requirements, design, code, or test cases.
* Test Environment - The hardware, software, and other resources required to conduct testing on a software application.
* Test Case - A set of steps, conditions, and expected results used to validate the functionality of a software application.
* Test Execution - The process of running test cases to validate the functionality of a software application.
* Test Metrics - Quantitative measurements used to evaluate the effectiveness and efficiency of the testing process.
* Sign-off - Formal approval by stakeholders or management indicating that testing has been completed and the software application is ready for release.
* Acceptance Criteria - A set of predefined criteria that a software application must meet in order to be accepted by stakeholders or end-users.
* Integration testing: A type of testing where individual components of a system are combined and tested as a group to ensure they work together correctly.
* Regression testing: A type of testing that involves retesting previously tested functionality to ensure that changes or updates to the system have not introduced new defects.
* Smoke testing: A type of testing that verifies that the basic functionality of an application is working as expected before more extensive testing is performed.
* Test case: A set of steps and expected outcomes that are designed to test a specific feature or functionality of an application.
* Test cycle: The process of planning, executing, and evaluating a set of tests.
* Test environment: The hardware, software, and network configurations used to perform testing.
* Test harness: A collection of software tools and utilities that are used to test an application.
* Test plan: A document that outlines the objectives, scope, approach, and schedule for testing an application.
* Test suite: A collection of test cases that are designed to test a specific area or component of an application.
* Usability testing: A type of testing that evaluates how easy it is for users to interact with an application and perform tasks.